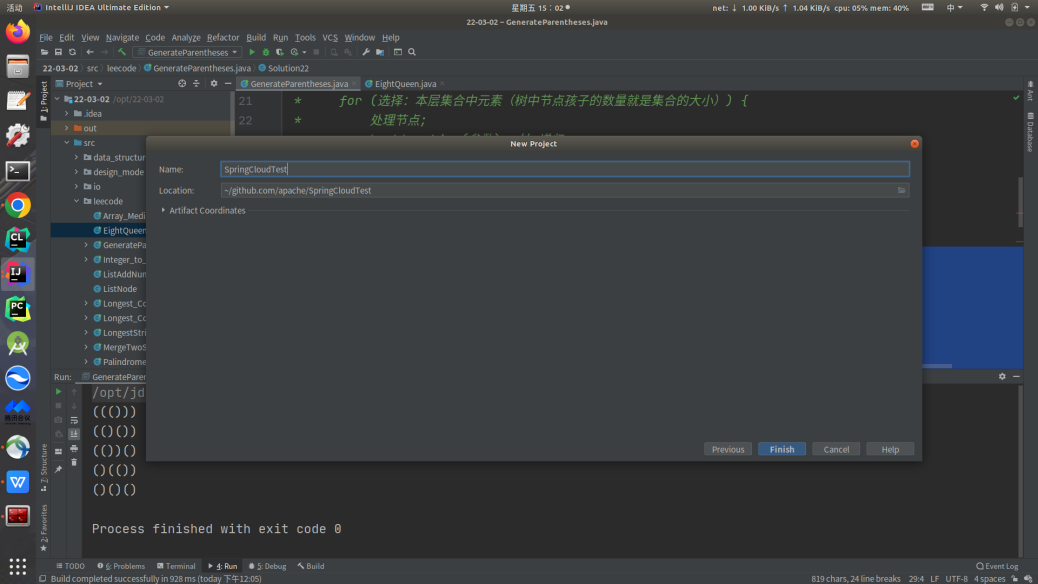
**SpringCloud:学习**

首先学会创建最基本的SpringCloud项目：

手顺如下

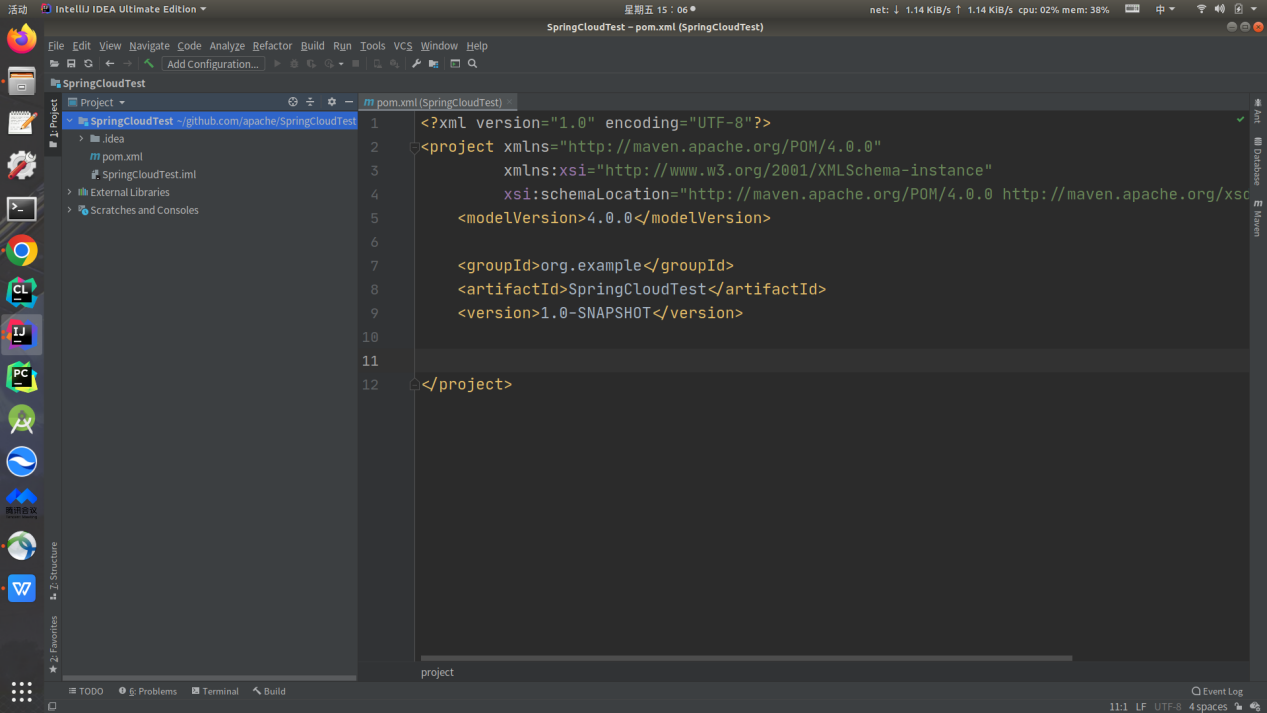
**第一步**

**创建一个MAVEN普通项目**

****

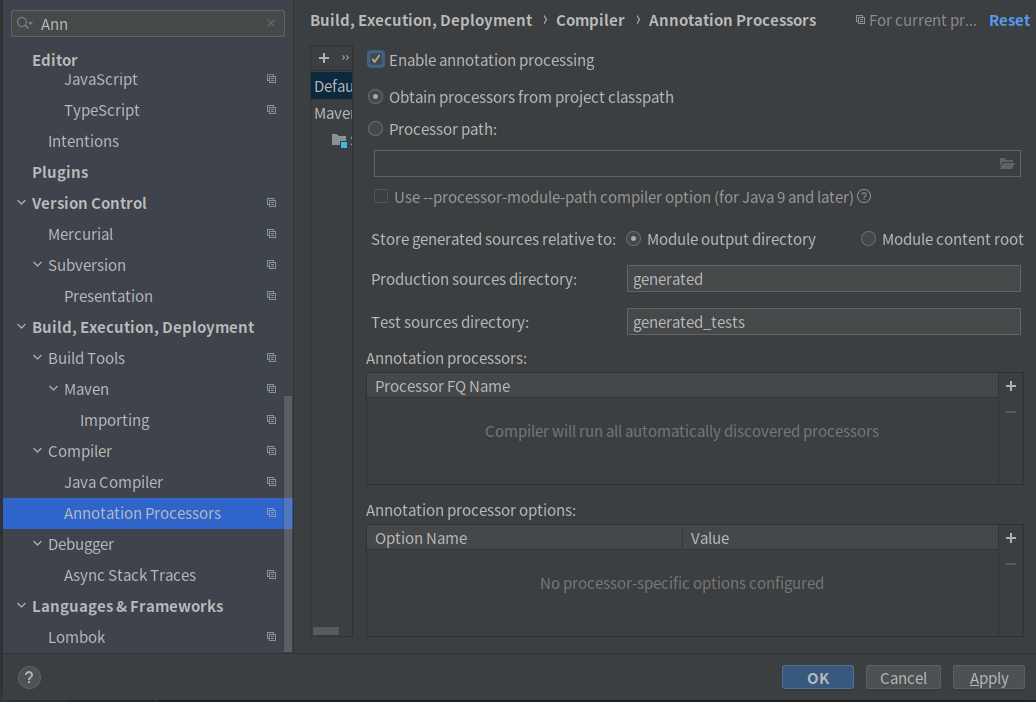
**第二步**

**删除里面的src文件夹**

****

**第三步**

**激活注解->从项目类路径中获取注释处理器并指定输出目录**

****

**第四步**

**加入以下基本依赖**

<!--Maven 项目查看maven依赖的详细信息。 -->

<!--在target的文件夹下，生成一个site文件夹，里边有个dependency.html，进去就能看到详细的依赖的信息-->

<dependency>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-project-info-reports-plugin</artifactId>

<version>3.0.0</version>

</dependency>

<!-- 下面三个基本是微服务架构的标配 -->

<!--spring boot 2.2.2-->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-dependencies</artifactId>

<version>2.2.2.RELEASE</version>

<type>pom</type>

<scope>import</scope>

</dependency>

<!--spring cloud Hoxton.SR1-->

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-dependencies</artifactId>

<version>Hoxton.SR1</version>

<type>pom</type>

<scope>import</scope>

</dependency>

<!--spring cloud 阿里巴巴-->

<dependency>

<groupId>com.alibaba.cloud</groupId>

<artifactId>spring-cloud-alibaba-dependencies</artifactId>

<version>2.1.0.RELEASE</version>

<type>pom</type>

<scope>import</scope>

</dependency>

<!--mysql-->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>${mysql.version}</version>

<scope>runtime</scope>

</dependency>

<!-- druid-->

<dependency>

<groupId>com.alibaba</groupId>

<artifactId>druid</artifactId>

<version>${druid.version}</version>

</dependency>

<dependency>

<groupId>org.mybatis.spring.boot</groupId>

<artifactId>mybatis-spring-boot-starter</artifactId>

<version>${mybatis.spring.boot.version}</version>

</dependency>

<!--junit-->

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>${junit.version}</version>

</dependency>

<!--log4j-->

<dependency>

<groupId>log4j</groupId>

<artifactId>log4j</artifactId>

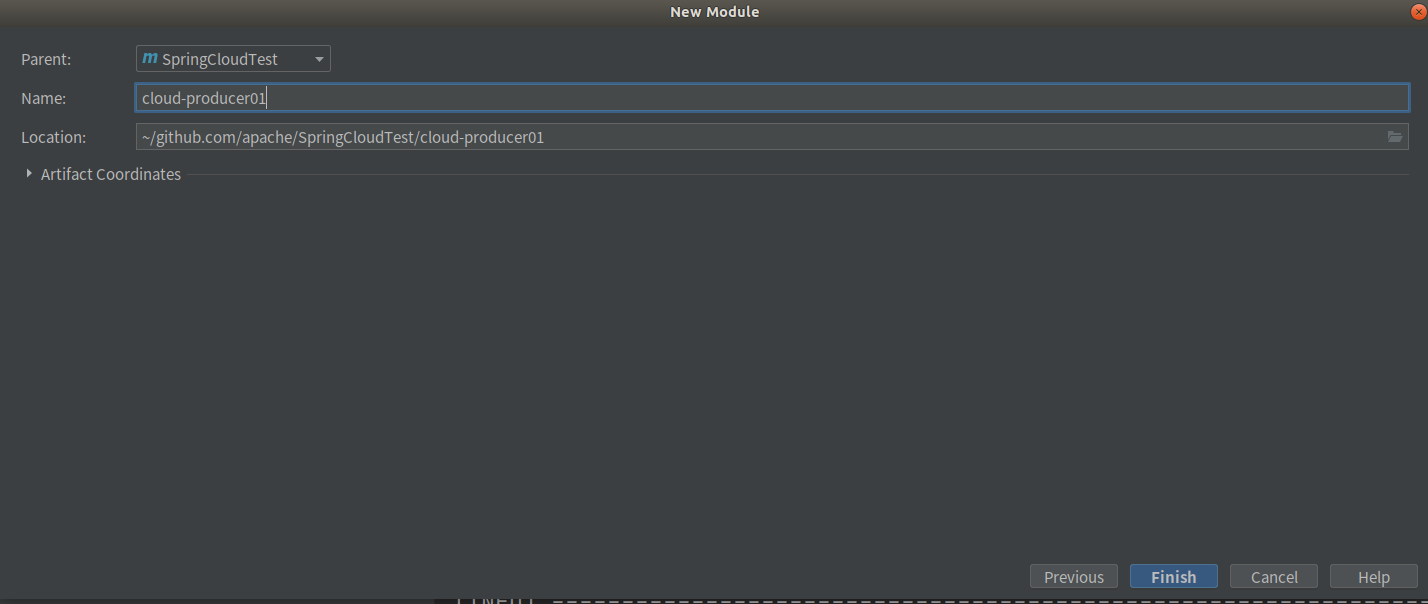
<version>${log4j.version}</version>

</dependency>

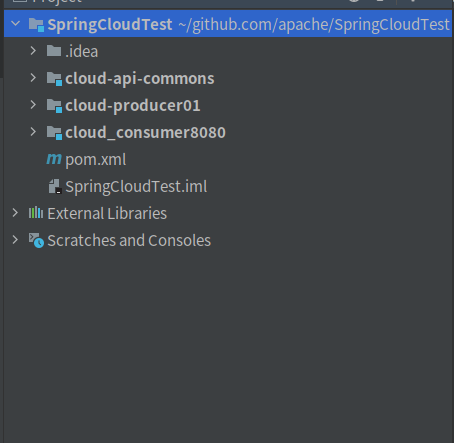
</dependencies>

**第五步**

**创建子项目**

****

**创建成这个样子**

****

结构说明：这是小试牛刀的一个简单小例子，只包含了生产者（cloud-producer01）消费者（cloud\_consumer8080）和项目恢复重构（cloud-api-commons）

**编写**cloud-api-commons**子项目：**

**在pom文件中输入以下依赖**

**<dependency>**

**<groupId>org.projectlombok</groupId>**

**<artifactId>lombok</artifactId>**

**</dependency>**

**<dependency>**

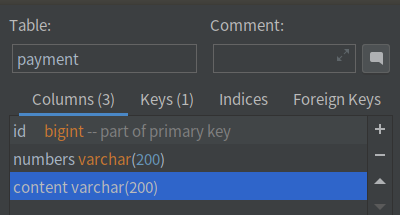
**<groupId>cn.hutool</groupId>**

**<artifactId>hutool-all</artifactId>**

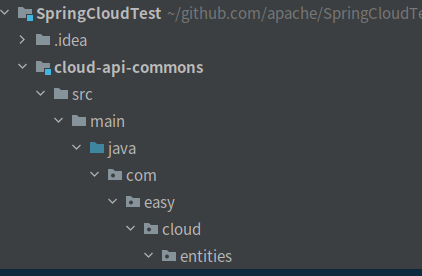
**<version>5.1.0</version>**

**</dependency>**

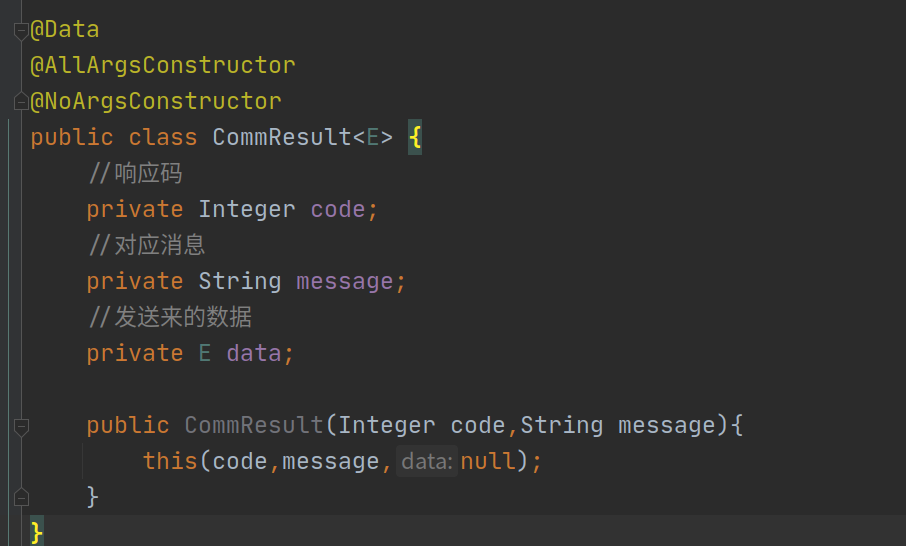
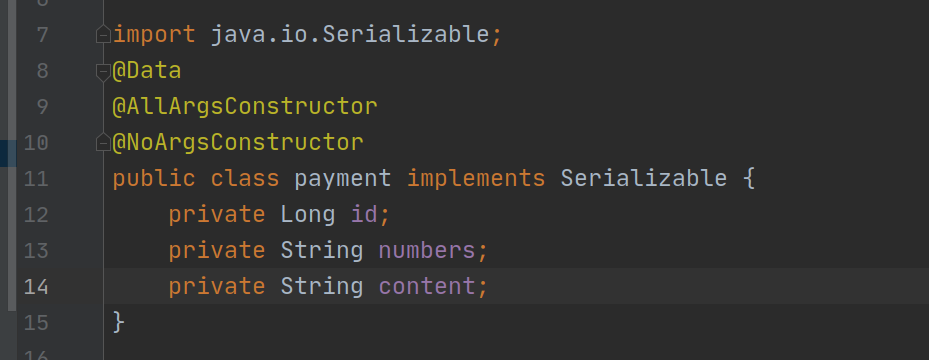
**创建一个订单表**

****

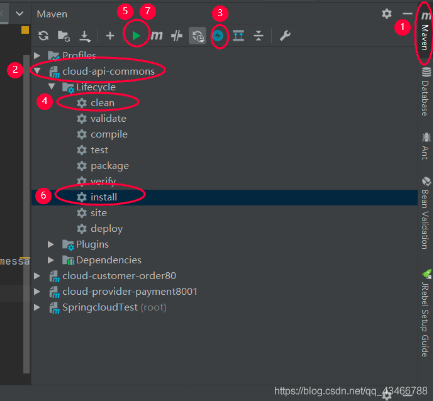
**创建一个基类package**

****

**创造基础类型payment(和数据库对应)CommResult(传输消息体)**

****

**然后按照步骤执行**

****

**链接上mysql**

**SpringCIoudNetfix：**

官方文档：https://spring.io/projects/spring-cloud-netflix

功能：

服务发现：可以注册 Eureka 实例，客户端可以使用 Spring 管理的 bean 发现实例

服务发现：可以使用声明性 Java 配置创建嵌入式 Eureka 服务器

1. **Archalus**

官方文档:https://www.baeldung.com/netflix-archaius-spring-cloud-integration

首先Archalus是一个强大的配置管理库，可用于从许多不同来源收集配置属性，提供对它们的快速、线程安全的访问。

**Archaius 提供了一些其他任何配置框架都没有考虑到的方便且有趣的功能.**

动态和类型属性

在属性突变时调用的回调机制

动态配置源（例如 URL、JDBC 和 Amazon DynamoDB）的即用型实施

可由 Spring Boot Actuator 或 JConsole 访问以检查和操作属性的 JMX MBean

动态属性验证